HTML-elementen

Met de *markeertaal* HTML bouw je webpagina's en kan je ze eveneens aan elkaar koppelen (linken). Een webpagina bouw je niet met een tekstverwerker. Je schrijft hem best in code. Een browser (Chrome, Firefox, Internet Explorer, Opera, Safari…) leest die code en zet die om in een weergave voor de mens. Je moet in je HTML-code precies aangeven waar een titel, alinea, link, afbeelding… of wat dan ook staat, want een browser is een stuk software en die software weet anders niet wat voor jou een titel of een alinea is.
Daarom moet je elk onderdeel "selecteren" en aan de browser zeggen: "Kijk, dit is een titel!" Het lijkt een beetje op het studeren van een tekst: je duidt de belangrijkste onderdelen met een markeerstift aan. We kunnen natuurlijk niets met een markeerstift beginnen als we een webpagina bouwen. In HTML (hypertext ***markup*** language) markeren we een element door aan te duiden waar het element begint en waar het eindigt. Dit doe je op de volgende manier:

<element>De inhoud van het element</element>

Natuurlijk moet je niet "element" invoeren, maar de afgesproken benamingen, meestal afkortingen: zo staat "p" voor "alinea" en "h1" voor een grote "kop".

In een webpagina zitten een aantal HTML-elementen. Je hoeft heus niet alles te kennen. Met een beperkt aantal basisonderdelen bouw je zo goed als alle mogelijke webpagina's.

Op de volgende pagina's vind je een overzicht van de belangrijkste HTML-elementen.

Titels	
kop 1	mijn titel
kop 2	mijn titel
kop 3	mijn titel
kop 4	mijn titel
kop 5	mijn titel
kop 6	mijn titel
groep titels (HTML5)	<hgroup></hgroup>
Andere tekstelementen	
alinea	Een tekst in een alinea.
artikel (enkel HTML5)	<article></article>
Sectie of rubriek (enkel HTML5): verzameling van artikels.	<section></section>
Voorbeeld	<section> <article> Titel van het artikel Eerste alinea Tweede alinea </article> <article> Titel van het tweede artikel Eerste alinea Tweede alinea </article> </section>
citaat	<blockquote>Tekst citaat</blockquote>
Lijsten	
genummerde lijst	...
niet-genummerde lijst	...
een item in de lijst	Appel
Voorbeeld	- Appel - Peer - Appel - Peer
Koppelingen	
hyperlink	Blabla
afbeelding	
video (HTML5)	<video src=”map/naamvideobestand.mp4”/>
audio (HTML5)	<audio src=”map/naamaudiobestand.mp3”/>

*Formulierelementen nemen we niet op, dat is al iets meer gevorderd.

Blokelementen en onderverdelingen	
Onderverdelingen	
Hoofdbalk, bannergedeelte (html5)	<header></header>
Onderdeel “naast” de hoofdinhoud (html5)	<aside></aside>
Navigatiebalk (html5)	<nav></nav>
Tekstonderdeel van de webpagina (html5)	<section></section>
Voetgedeelte (html5)	<footer></footer>

Een lekker nestje

In HTML kan je elementen “nesten”. Dit wil zeggen dat het ene element in het andere kan zitten. Net zoals je in een grote doos kleinere dozen en in die kleinere dozen nog kleinere of losse elementen. Je moet die elementen dan ook in elkaar steken en niet over elkaar laten hangen.

Zo kan je in een alinea bijvoorbeeld een stukje vetgedrukt maken:

Dit is een tekst met een paar zeer belangrijke woorden in.

De HTML-markering (HTML-tag) duidt aan dat (een stukje van) de tekst vetgedrukt moet weergegeven worden. Of je kan nog dieper “nesten” en een deel van die belangrijke woorden ook “schuingedrukt” weergeven.

Dit is een tekst met een paar zeer belangrijke woorden in.

Bij een navigatielijst wordt dit zeer duidelijk:

1. We maken eerst een niet-genummerde lijst:
2. In dit lijst plaatsen we lijstitems.
3. In elk lijstitem plaatsen we een link.
4. In elke link plaatsen we een opschrift waarop de gebruiker kan klikken.

```
<ul>
<li><a href=”http://www.eensitewaarwenaarlinken.be”>Opschrift met de naam van de site</a></li>
<li><a href=”http://www.eenanderesite.be”>Nog een opschrift</a></li>
</ul>
```

Verschil tussen HTML/XHTML en HTML5

In HTML5 zijn een aantal nieuwe elementen geïntroduceerd. De bedoeling was om

- overbodige elementen weg te gooien
- een aantal praktijken van webdesigners te standaardiseren
- een aantal dingen te vereenvoudigen
- nieuwe mogelijkheden te bieden (modernere)
- html beter doorzoekbaar te maken voor zoekmachines (semantische webpagina's)

Een voorbeeld in HTML/XHTML:

Je kan aan elke element een id (eigen identiteit) geven. *De naam die je aan zo'n id geeft, mag je zelf kiezen*. Veel webdesigners kozen ongeveer dezelfde namen. Zo wisten ze zelf waarvoor elk onderdeel diende. Maar zoekmachines wisten bijvoorbeeld niet waar de navigatiebalk of de eigenlijke inhoud stond omdat er geen “standaard” was.

```
<body>
	<div id=”container”>
		<div id=”header”>
		</div>
		<div id=”nav”>
		</div>
		<div id=”content”>
			<p>Dit is een tekst bij een foto.</p>
			<img src=”foto.jpg” title=”Mijn mooiste foto”/>
		</div>
		<div id=”footer”>
		</div>
	</div>
</body>
```

Een voorbeeld in HTML5:

In HTML5 gebruik je nog steeds waar nodig een id. Maar i.p.v. voor zoekmachines onduidelijke “div”-elementen, *weten zoekmachines nu meteen waarvoor elk onderdeel dient*.

```
<body>
	<div id=”container”>
		<header>
		</header>
		<nav>
		</nav>
		<section>
			<article>
				<figure>
					<figcaption>Dit is een tekst bij een foto.</figcaption>
					<img src=”foto.jpg” title=”Mijn mooiste foto”/>
				</figure>
			</article>
		</section>
		<footer>
		</footer>
	</div>
</body>
```

Een webpagina is een doos met andere dozen en onderdelen in

Een HTML-pagina is een doos. Laat het ons het even voorstellen met een analogie uit de werkelijkheid. Beeld je het volgende in…

> Je krijgt thuis een doos (een <html>-pagina) toe waarin zich alle onderdelen van een kast bevinden. De doos wordt geleverd door een transportfirma (internetverbinding). Je krijgt ook een leveringsnota met daarin de naam van de verzender, de titel van het product en nog een boel algemene informatie. Die “leveringsnota” is het <head>-gedeelte van je webpagina. De doos zelf staat voor het <body>-element.
>
> Jij opent de doos om de kast samen te stellen (jij bent de browser). Als je de doos opent, zie je meteen dat er van boven naar beneden andere dozen insteken (-, <nav>-, <aside>- …. elementen). In die kleinere dozen zitten weer andere dozen en in die dozen zitten kleinere onderdelen. Jij moet ze nu op de juiste manier in elkaar passen en samenstellen. Gelukkig heb je een apart document met daarin een handleiding (een CSS-bestand). In zo'n handleiding staat precies waar elk onderdeel moet staan en hoe het er moet uitzien. Bijvoorbeeld: het ene onderdeel leg je links, maar moet je eerst nog rood schilderen, het andere onderaan enz.
>
> Weet je, je wil nu meerdere van diezelfde kasten, want het oogt niet mooi als je tien verschillende types kasten in één kamer (=1 website) plaatst. Je bestelt dus meerdere dozen. Gelukkig kan je voor al die kasten dezelfde handleiding gebruiken. Je moet die handleiding ook maar één keer bestellen (op de server), want je hebt die nog van de vorige keer. Tenzij je de handleiding kwijtgespeeld bent (de cache van je browser leeggemaakt), dan moet je ze opnieuw gaan opvragen en laten opsturen.
>
> Nu weet je perfect hoe je de kast(en) moet opbouwen. En als je de handleiding nog hebt, verloopt de opbouw sneller dan de eerste keer.
> Op de leggers (titels, alinea's, <img>-elementen…) plaats je teksten, foto-albums, videocassettes enz. tot je kasten netjes gevuld zijn. Voor de bezoekers (je hebt bijvoorbeeld een winkel) maak je nu een reeks wegwijzers zodat iedereen weet waar wat te vinden is (een <nav>-gedeelte met hyperlinks).

HTML en CSS

Met HTML alleen kan je een webpagina wel opbouwen en van inhoud voorzien, maar je kan er niet mee bepalen waar welke onderdeel precies moet staan. Je kan er ook het lettertype, de kleuren, de breedte, de hoogte enz. niet mee instellen. Kortom… vergeet opmaak. Maar dit klopt niet helemaal.
Je moet er als browser uiteindelijk wel *iets* mee doen. Zonder opmaak toont de browser de onderdelen zoals hij ze van boven naar beneden in de HTML-code vindt.

> Denk opnieuw aan het verhaal van de doos en de kast. Wat een browser standaard met HTML doet, lijkt een beetje op met een breekmes de kartonnen doos opensnijden waarin de onderdelen van de kast steken. Dan kijk je uit op alle onderdelen van boven naar beneden, maar alles zit nog niet op zijn plaats. De onderdelen zijn nog niet geschilderd enz. De browser toont eerst het element dat hij bovenaan in de code vindt en hij leest zo door tot hij helemaal onderaan de pagina komt. Misschien liggen er drie planken onder elkaar die je liever naast elkaar zou zien staan. Zo gaat het ook met HTML in de browser. Zonder handleiding (CSS) ligt niets precies op zijn plaats. Je kan er alleen op rekenen dat de bovenste doos, de bovenste elementen van je kast bevat en de onderste doos de onderste elementen.

Tussenliggend besluit: een browser leest de HTML-elementen van boven naar beneden in. Wat bovenaan in de code staat, bekijkt hij het eerst.

Maar ik hoor u al denken… als ik geen lettertype of lettergrootte kan instellen, hoe komt het dan dat ik de tekst toch kan lezen? Waarom is een link dan blauw en onderstreept? Tja, een browser moet natuurlijk wel iets doen met de inhoud, hij moet iets tonen. Een browser kiest een soort standaard opmaak. Meestal is dit Times New Roman als lettertype. De achtergrond is standaard wit met zwarte letters. Links worden voor de duidelijkheid blauw en onderstreept weergegeven.

Tussenliggend besluit: een browser kiest een standaard opmaak als je geen CSS voorziet.

Herkennen alle browsers alle HTML-elementen? Neen, oudere browsers (en vaak ook oudere versies van Microsoft Internet Explorer) herkennen bepaalde elementen nog niet omdat ze pas recent zijn toegevoegd. Oudere browser herkennen de nieuwe HTML5-elementen niet. Oeps, wat gebeurt er dan? Als ze gewoon “tekst” bevatten, interpreteren oudere browsers die elementen gewoon als tekstelementen.

Bijvoorbeeld:
<nav>Dit is tekst</nav>

zal in oudere versies van Internet Explorer worden geïnterpreteerd als * Dit is tekst * of *Dit is tekst*.

Als de browser echt geen raad weet met het element (bijvoorbeeld met de <video>-markering), toont hij gewoon… niets!

HACKS of WORKAROUNDS

Maar daar bestaan natuurlijk oplossingen voor. Webdesigners noemen zo'n oplossingen workarounds (we draaien een beetje rond de pot om onze goesting te krijgen) of hacks.

Zulke hacks zijn vaak slimme stukjes code in javascript of CSS of bepaalde technieken die enkel Internet Explorer ondersteunt. **

Op die manier kunnen we de meeste onbekende elementen toch correct laten weergeven of omzeilen met een “alternatief” voor die specifieke browser.

** De meeste browsers ondersteunen afgesproken standaarden. Microsoft ontwikkelde in het verleden vaak zijn eigen alternatieven om extra mogelijkheden te bieden of om te proberen hun eigen wil door te drijven.

Opmaak met CSS

CSS is een opmaaktaal. CSS ziet er compleet anders uit dan HTML. Maar het principe is hetzelfde.

1. Je selecteert een element of een onderdeel daarvan.
2. Je bepaalt welke van de onderstaande stijlregels op dit element van toepassing zijn.
3. Je bewaart al die stijlregels in een apart document/bestand.
4. Je “linkt” dit CSS-bestand aan al je webpagina's.
5. Al je webpagina's hebben nu dezelfde stijl. Bijvoorbeeld: alle titels zijn even groot, hebben dezelfde kleur enz.

Wat bedoelen we met stijlregels?

Tekstopmaak:

- Lettertype
- Lettergrootte
- Kleur
- Kapitalen
- Tekstschaduw
- Uitlijning
- …

Opmaak van blokken:

- Breedte
- Hoogte
- Schaduw
- Rand
- Rotatie
- Afronding
- Zichtbaarheid
- …

Positie van elementen:

- Positie ten opzichte van ander element
- Afstand van bovenzijde
- Afstand van linkerzijde
- Marge
- Padding
- …

We gaan van start…

Bovenaan (in de <head>-sectie van je HTML-document) voeg je de volgende markering toe:

```
<style>


</style>
```

Hier tussen schrijven we al onze stijlregels. Opgelet: verder op zullen we zien dat je die stijlregels best in een afzonderlijk document bewaart. Maar dat sparen we nog even voor later…

Elementen selecteren met CSS

CSS ziet er anders uit dan HTML. Met CSS (cascading style sheet) moet je een HTML-element selecteren en er vervolgens bepaalde stijlen aan toekennen. Klinkt dat een beetje Chinees? Het wordt wel duidelijk. Selecteren doe je in dit geval niet door over een stuk tekst te slepen met de muis. Je selecteert elementen met CSS-code. Je kan dit op verschillende manieren:

1. De ganse pagina selecteren.
2. Bepaalde HTML-elementen selecteren.
3. Eén enkel element selecteren.
4. Een *select* groepje selecteren.

1. De ganse pagina selecteren

Bij CSS helpt een grote portie gezond boerenverstand. Stijlregels die voor alle onderdelen van je pagina gelden, schrijf je maar één keer. Als je bijvoorbeeld voor de meeste of alle elementen hetzelfde lettertype wil gebruiken, schrijf je dit maar één keer. Hiervoor moeten we eerst alle elementen selecteren. Alle elementen selecteren doe je op de volgende manier:

```
<style>
*{}
</style>
```

of

```
<style>
html{}
</style>
```

of

```
<style>
html,*{}
</style>
```

2. Bepaalde HTML-elementen selecteren.

Standaard beeldt een browser links of koppelingen *blauw* en *onderstreept* af. Dat oogt niet altijd mooi. Als je dit wil wijzigen, moet je eerst alle links op de webpagina selecteren en er vervolgens de stijl van aanpassen.

Een link stop je in HTML in een <a>-markering. Dit betekent dat we met CSS alle <a>-elementen moeten selecteren.

```
<style>
a{}
</style>
```

Je kan natuurlijk ook meerdere elementen selecteren. Dit doe je onder elkaar:

```
<style>
*{}
body{}
a{}
p{}
</style>
```

3. Eén enkel element selecteren.

Wil je alle links in het groen, maar één bepaalde link in het geel? Wil je rond één bepaalde alinea (<p>) een rand plaatsen van 1 pixel breed? Wil je de banner een bepaalde hoogte en breedte geven? Natuurlijk kan dit. Maar dit betekent wel zoiets als zeggen: alle leerlingen moeten een groen vestje dragen, maar Joske moet een geel vestje dragen. Dit betekent dat we het element dat we een apart uitzicht willen geven, ook een identiteit (id) moeten toekennen.

In HTML ziet het er dan bijvoorbeeld als volgt uit:

```
<p>Dit is een alinea.</p>
<p>Dit is nog een alinea.</p>
<p>Dit is nog een andere alinea.</p>
<p id="joske">Dit is een alinea die ik een apart uitzicht wil geven.</p>
```

Als we een element op zijn "id" willen selecteren (in het bovenstaande geval 'joske') dan moeten we in CSS een #-teken plaatsen, gevolgd door het "id".

```
<style>
p{}
#joske{}
</style>
```

of

```
<style>
p{}
#joske{}
</style>
```

4. Een *select* groepje selecteren.

Wat als je nu niet één, maar enkele elementen wil selecteren? Alsof je in een klasje zou zeggen: "Alle kindjes met blond haar gaan aan de linkerkant staan." Het wordt wellicht nog duidelijker wanneer we het vergelijken met auto's. Bijvoorbeeld: Alle personenauto's hebben een groene carrosserie, behalve de Mercedes klasse 190.

Ook dit kan... maar dan moet je aan de elementen waaraan je een gelijkaardig uitzicht wil geven een "klasse" (class) toekennen.

In HTML ziet dit er bijvoorbeeld als volgt uit:

```
<p>Dit is een alinea.</p>
<p class="apart">Dit een alinea met een aparte stijl.</p>
<div class="apart">Dit is een blokelement met een aparte stijl.</div>
<a href="http://www.blabla.be" class="apart">Klik hier</a>
```

In CSS selecteer je een "klasse" door een . (punt) te schrijven, gevolgd door de naam van de klasse:

```
<style>
.apart{}
</style>
```

Stijlregels toepassen

Je kan aan één CSS-selectie meerdere stijlregels toekennen. Gebruik ook hier je gezond boerenverstand. Het heeft bijvoorbeeld geen zin om een lettertype in te stellen voor afbeeldingen.

De onderstaande code is ideaal om links af te beelden in een formaat dat ideaal is voor mobiele browsers. Het maakt de links zwart, verwijdert de streep onder de link. De regel "display:block" zorgt ervoor dat de link in de browser wordt weergegeven als een blokelement. Hierdoor kan je aan de link (ondanks zijn tekstinhoud) een vaste breedte en hoogte geven. Standaard is een link immers niet breder dan de lengte van de linktekst. Staat er als opschrift voor je link bijvoorbeeld "Klik hier", dan kan je enkel op "Klik hier" klikken. Met het onderstaande voorbeeld komt er een virtueel kader met 100% in je webpagina. De linktekst blijft even breed, maar de linktekst staat nu in een groot aanklikbaar vlak.

```
<style>
a{
        color:black;
        text-decoration:none;
        display:block;
        width:100%;
        height:30px;
        background-color:#ccc;
        border:1px solid black;
}
</style>
```

Op webpagina's merk je vaak "kolommen". Toch is het maken van kolommen niet iets dat standaard kan met CSS. In de nieuwste versie van CSS zou dit moeten kunnen, maar jammer genoeg werkt dit in heel wat browsers (lees: Internet Explorer) nog niet en kunnen we er dus niet op rekenen. Wat moet je dan doen als je bijvoorbeeld een navigatiebalk aan de linkerkant van je pagina wil plaatsen en de inhoud rechts? Dit kan, met wat omwegen.

In HTML schrijven we:

```
<div id="container">
        <div id="nav">
                Dit is de navigatiebalk die links moet getoond worden.
        </div>
        <div id="content">
                Dit is de tekstinhoud van mijn webpagina.
        </div>
</div>
```

Je merkt dat het element met id "container" 2 andere -elementen bevat. Eerst een met id "nav", daarna een met id "content". Standaard plaatst een browser die onder elkaar. Met CSS kunnen we ze naast elkaar plaatsen.

```
<style>
#container{
        width:960px;
}
#nav{
        width:260px;
        float:left;
}
#content{
        margin-left:270px;
}
</style>
```

De regel float:left zorgt ervoor dat "nav" links uitlijnt en de regel width:260px zorgt ervoor dat dit element niet breder is dan 260 pixels.
Het -element "content" krijgt een linkermarge van 270 pixels waardoor het rechts naast "nav" gaat staan.

Met CSS kan je in beperkte mate ook "interactiviteit" toevoegen aan links. Als je bijvoorbeeld over een link beweegt, dan kan je het uitzicht van die link wijzigen:

```
<style>
a{
        color:black;
        text-decoration:none;
        display:block;
        width:100%;
        height:30px;
        background-color:white;
        border:1px solid black;
}
a:hover{
        color:white;
        background-color:black;
}
</style>
```

In het bovenstaande voorbeeld is de linktekst zwart met een witte achtergrond. Als de bezoeker van de webpagina met zijn/haar muis over de link beweegt, wordt de linktekst wit en de achtergrondkleur zwart.

CSS en HTML aan elkaar koppelen

In de bovenstaande voorbeelden hebben we de CSS-stijlregels rechtstreeks in de webpagina geplakt. Dat is niet zinvol, want we willen dezelfde stijl voor meerdere pagina's kunnen gebruiken. Hiervoor bewaren we alle regels in een afzonderlijk document met de uitgang ".css". We bewaren dit CSS-bestand eveneens op de server (of op een andere server). Belangrijk is dat we het correcte "adres" kennen, want nadien moeten we dit bestand aan al onze webpagina's koppelen.

In de bovenstaande voorbeelden plaatsten we al onze stijlregels tussen <style> en </style>.

```
<style>

</style>
```

Dit zijn HTML-markeringen. Vermits we nu alle regels in een apart document bewaren (dat bovendien niet is opgesteld in HTML, moeten we dit niet meer doen. Je maakt dus een nieuw document en je bewaart het bijvoorbeeld onder de naam stijl.css.

In dit bestand schrijf je meteen je stijlregels

```
p{}
a{}
```

Nu rest er nog één stap. We moeten dit stijldocument linken aan onze webpagina's. Hiervoor moeten we een <link>-element opnemen in de <head>-sectie van onze webpagina's:

```
<link rel="stylesheet" type="text/css" href="stijl.css"/>
```

Als je het CSS-bestand in een onderliggende map met bijvoorbeeld de naam "stijlen" hebt bewaard, dan moet je het "href"-attribuut aanpassen:

```
<link rel="stylesheet" type="text/css" href="stijlen/stijl.css"/>
```

Je kan ook een CSS-bestand van een andere website opnemen:

```
<link rel="stylesheet" type="text/css" href="http://www.blabla.be/css/mystyle.css">
```

CSS voor meerdere doeleinden

Het kan handig zijn om uw site een ander uitzicht te geven als hij afgedrukt wordt, wanneer hij via een beamer wordt geprojecteerd, wanneer een blinde de pagina bezoekt (en software de 'tekst voorleest'), wanneer de pagina met een mobiel toestel wordt bezocht enz.

Je kan voor elk van die situaties een afzonderlijk CSS-document opnemen (als je dit wenst). Hiervoor neem je een "media"-attribuut op in je <link>-element.

Media	Van toepassing op...
all	alle situaties
screen	schermweergave (computerscherm)
print	drukweergave
aural	voorleessoftware (voor blinden)
braille	brailletoestellen
embossed	brailletoestellen
projection	projectors, beamers (werkt enkel in Opera)
handheld	kleine draagbare toestellen (Opgelet: niet noodzakelijk op mobiele telefoons, hiervoor moet je "javascript" gebruiken.)
tv	televisieweergave

Dit houdt in dat je meerdere CSS-bestanden kan koppelen. Het heeft natuurlijk geen zin om in alle bestanden dezelfde inhoud te plaatsen.

```
<link rel="stylesheet" type="text/css" href="stijlen/algemeen.css" media="all"/>
<link rel="stylesheet" type="text/css" href="stijlen/druk.css" media="print"/>
<link rel="stylesheet" type="text/css" href="stijlen/scherm.css" media="screen"/>
<link rel="stylesheet" type="text/css" href="stijlen/projector.css" media="projector"/>
<link rel="stylesheet" type="text/css" href="stijlen/zicht.css" media="aural"/>
```

Opmerking: niet alle mediatypes worden door alle browsers ondersteund! Voor een overzicht van de compatibiliteit: http://www.codestyle.org/css/media/tty-BrowserSummary.shtml.

Een voorbeeld:

Je kan bij het afdrukken bepaalde elementen onzichtbaar maken. Zo heeft het geen zin om een navigatiebalk of een banner mee af te drukken. Bekijk het volgende stukje HTML. We hebben dit reeds eerder aangehaald.

```
<html>
<head>
	<link rel="stylesheet" type="text/css" href="stijlen/algemeen.css" media="all"/>
	<link rel="stylesheet" type="text/css" href="stijlen/scherm.css" media="all"/>
	<link rel="stylesheet" type="text/css" href="stijlen/druk.css" media="print"/>
</head>
<div id="container">
	<div id="banner">
		<h1>Hondenkapsalon Leuven</h1>
	</div>
	<div id="author">
		&copy; Kris Merckx, 2012, Alle rechten voorbehouden.
	</div>
	<div id="nav">
		Dit is de navigatiebalk die links moet getoond worden.
	</div>
	<div id="content">
		Dit is de tekstinhoud van mijn webpagina.
	</div>
</div>
</body>
</html>
```

Een deze webpagina zijn 3 CSS-bestanden gekoppeld: ***algemeen.css***, ***scherm.css*** en ***print.css***.

Inhoud van het bestand algemeen.css

In het bestand "algemeen.css" nemen we algemene stijlregels op die zowel van toepassing zijn voor schermweergave als voor drukweergave.

```
html, *{
        font-family:Arial, helvetica, sans-serif;
        color:black;
}
a{
        color:black;
        text-decoration:none;
        display:block;
        width:100%;
        height:30px;
        background-color:white;
        border:1px solid black;
}
a:hover{
        color:white;
        background-color:black;
}
```

Inhoud van het bestand scherm.css

Bij de schermweergave tonen we de "container" met een breedte van 960 pixels. In die container vinden we onder elkaar twee "rijen". De eerste rij is de "banner" met een hoogte van 180 pixels. In die banner plaatsen we een achtergrondafbeelding (banner.jpg) die slechts één keer wordt weergegeven. Je zorgt dat die afbeelding 960 pixels bij 180 pixels meet.
De titel die in de banner staat maken we "onzichtbaar". Google of een andere zoekmachine zal die titel wel "uitlezen", maar de gewone gebruiker krijgt de achtergrondafbeelding te zien. Ook het -element met id "author" maken we onzichtbaar op een computerscherm. Dit zou er dan als volgt moeten uitzien.

Maak het <h1>-element in de met id "banner" onzichtbaar:

```
#banner h1{
        display:none;
}
```

Maak het -element met id "author" onzichtbaar:

```
#author{
        display:none;
}
```

Maar we kunnen hiervoor ook een verkorte schrijfwijze gebruiken. Vermits 2 elementen dezelfde stijlregel toegewezen krijgen, moeten we dit maar één keer schrijven:

```
#banner h1, #author{
        display:none;
}
```

Op een computerscherm krijgt de bezoeker dus eerst de banner te zien en daar onder naast elkaar de met id "nav" aan de linkerkant en de met id "content" aan de rechterkant (dit hebben we al eerder uitgelegd):

```
body{
        background-color:#ccc;
}
#container{
        width:960px;
}
#banner{
        height:180px;
        background-image:url(banner.jpg);
        background-repeat:no-repeat;
        background-position: top left;
        background-attachment:fixed;
}
#banner h1, #author{
        display:none;
}
#nav{
        width:260px;
        float:left;
}
#content{
        margin-left:270px;
}
```

Inhoud van het bestand print.css

We hebben ook een CSS-bestand gekoppeld dat wordt opgeroepen als de bezoeker onze webpagina wil afdrukken. In dat geval

- maken we de "banner" en het navigatiegedeelte onzichtbaar,
- laten we de met id "content" de volledige breedte van de pagina innemen
- en tonen we de met id "author".

Vermits "screen.css" bij het afdrukken niet wordt opgeroepen, is geen enkel van de regels in dat document van toepassing. In het bestand "print.css" nemen we nu de volgende regels op:

```
#banner, #nav{
        display:none;
}
#author{
        border:1px solid black;
        font-variant:small-caps;
}
body{
        background-color:white !important;
        color:black !important;
}
```

We zorgen dat het <body>-element een witte achtergrond krijgt en het lettertype zwart is. Dit bespaart de gebruik vooral ook inkt. De toevoeging "!important" zorgt ervoor dat eventuele andere instellingen in het bestand "algemeen.css" worden overschreven door deze nieuwe waarden.

Opgelet: standaard drukt de printer sowieso geen achtergrondafbeeldingen af.

Interessante links en goede leerschool

Interessante links voor html, css, javascript… :

- http://www.w3schools.com
- http://www.computerkit.be/cursus

Webapplicaties (Kris Merckx)

Uitgeverij: SDU Academic Service

Jaar van uitgave: 2010

ISBN-nummer: ISBN 9789012582858

Uitgave/druk: 1

Onderwerpen/tags: Toegankelijke gebruikersinterfaces bouwen voor het web

HTML5, Javascript, jQuery, webframeworks, SVG en Canvas maken het mogelijk om webapplicaties te bouwen die bijna niet meer te onderscheiden zijn van klassieke desktopprogramma's. Vaak zijn ze zelfs flexibeler, crossplatform en crossbrowser.

Dankzij een aantal open source frameworks kan iedere applicatieontwikkelaar met een basiskennis van HTML, Javascript en CSS in een mum van tijd zijn eigen killer-webapp bouwen. Het boek Webapplicaties van webontwikkelaar en auteur **Kris Merckx** richt zich zowel op klassieke browsers als Firefox, Internet Explorer en Chrome als op platformen voor smartphones en mobiele devices (iOS, Android, Blackberry). Dit praktische boek gaat onder andere in op Office-applicaties, media, tekenmogelijkheden, animaties, games en lokalisatie.